Introduction

Table of Contents

Data Preparation 1
Algorithm 1
Analysis 2

The USPS Handwritten Digit database is a set of handwriting data containing 1100 examples of the digits 0-9. To process and learn this data, a logistic regression algorithm is implemented.

Data Preparation

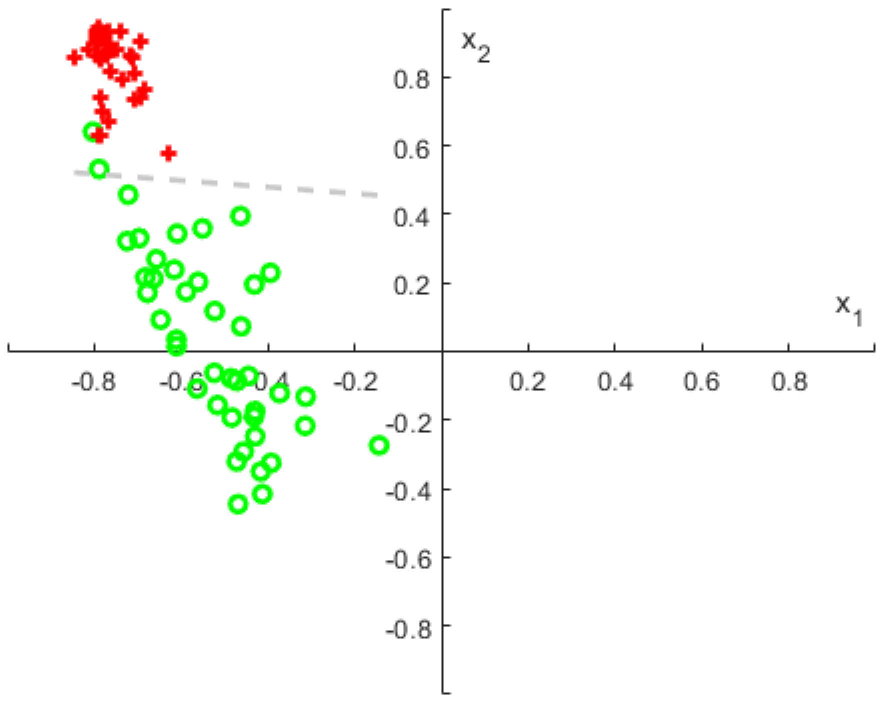
To prepare for training, data is pulled from the USPS handwriting set and processed. `getfeatures()` is called to find the symmetry and intensity of each data point. The resulting set is split into a training and test set.

```
load('usps_modified.mat');
[xd,yd] = getfeatures(data);
for i = 501:height(yd)
    yd(i) = -1;
end
N = 40;
[x_t,y_t,x,y] = split_data(xd,yd,N);
```

Algorithm

Both the stochastic and batch algorithms perform fairly well. Their iterations are limited by both a maximum amount of iterations and the gradient. If the gradient is too small, the algorithm will consider itself complete. This method was chosen because it's quite simple, which keeps the runtime low. Low runtimes allow for more iterations or even multiple layers in the future.

```
w = reg_bat(x_t, y_t, 10);
g_plot(x_t, y_t, w);
```

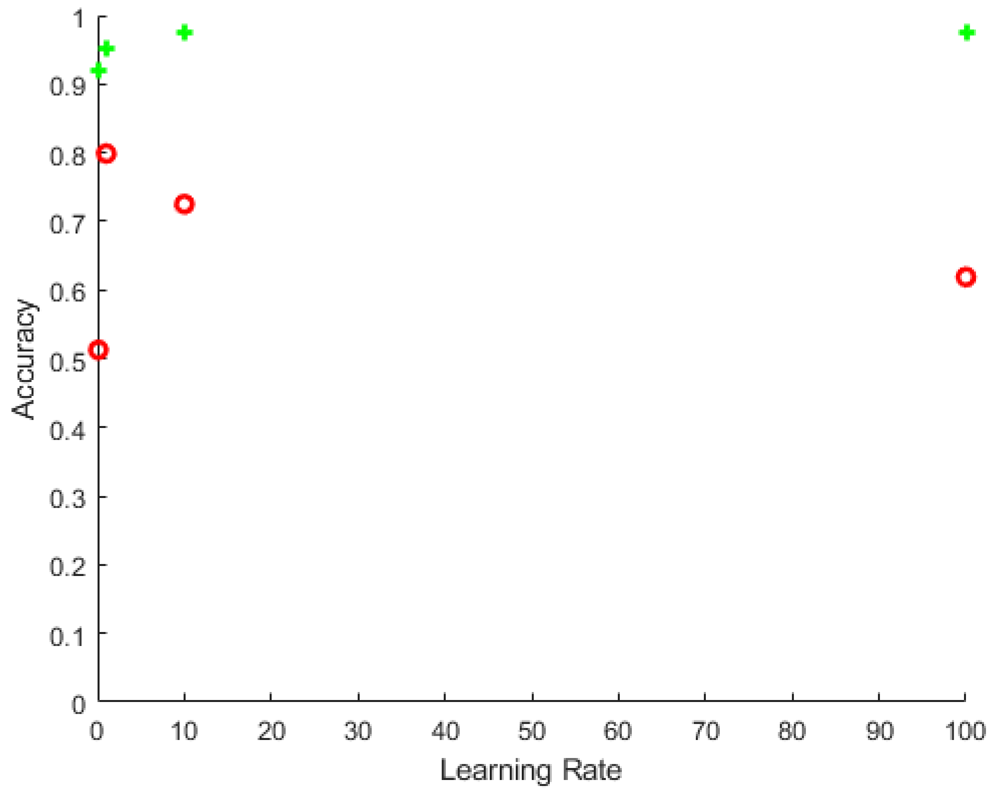


Analysis

The algorithm was run 10 times for each value of η. Learning rate affects how quickly the line will adapt to gradients. The accuracy of my implementation seems to greatly improve as the learning rate does. This might suggest that the algorithm stops too soon or that 0.1 is too small a learning rate. Another issue is that the Stochastic regression has lower accuracy. From tests, it also seems that it just moves towards the ideal too slowly.

```
d_sto = zeros(10,2,4);
d_bat = zeros(10,2,4);
for i = 1:10
    [x_t,y_t,x,y] = split_data(xd,yd,N);
    for eta = 1:4
        w_sto = reg_sto(x_t, y_t, power(10,eta-2));
        d_sto(i,1,eta) = g_test(x_t, y_t, w_sto);
        d_sto(i,2,eta) = g_test(x,y,w_sto);
        w_bat = reg_bat(x_t, y_t, power(10,eta-2));
        d_bat(i,1,eta) = g_test(x_t, y_t, w_bat);
        d_bat(i,2,eta) = g_test(x,y,w_bat);
    end
end
sto_avg = [mean(d_sto(:,1,:)) mean(d_sto(:,2,:))];
bat_avg = [mean(d_bat(:,1,:)) mean(d_bat(:,2,:))];
figure();
hold on;
axis([0 100 0 1])
```

```
xlabel('Learning Rate');
ylabel('Accuracy');
ax = gca;
ax.XAxisLocation='origin';
ax.YAxisLocation='origin';
for e = 1:4
    scatter(10^(e-2),mean(d_sto(:,1,e)),'r','O','LineWidth',2);
    scatter(10^(e-2),mean(d_bat(:,1,e)),'g','+','LineWidth',2);
end
```



Published with MATLAB® R2021a